CGMP: Cloud-assisted Green Multimedia Processing

[1,2]Yujun Ma,[3]Yin Zhang, [4]Zhengguo Sheng, [5]Zohreh Sanaei, [1]Min Chen
[1]School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
[2]Computer Network Center, Nanyang Institute of Technology, Nanyang, China
[3]School of Information and Safety Engineering, Zhongnan University of Economics and Law, Wuhan, China
[4]School of Engineering and Informatics, University of Sussex, UK
[5]Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia.
Email: yujun.hust@gmail.com, yin.zhang.cn@ieee.org, z.sheng@sussex.ac.uk, sanaei@ieee.org, minchen2012@hust.edu.cn

***Abstract*—With continued advancements of mobile computing and communications, emerging novel multimedia services and applications have attracted lots of attention and been developed for mobile users, such as mobile social network, mobile cloud medical treatment, mobile cloud game. However, because of limited resources on mobile terminals, it is of great challenge to improve the energy efficiency of multimedia services. In this paper, we propose a cloud-assisted green multimedia processing architecture (CGMP) based on mobile cloud computing. Specifically, the tasks of multimedia processing with energy-extensive consumption can be offloaded to the cloud, and the face recognition algorithm with improved principal component analysis and nearest neighbor classifier is realized on CGMP based cloud platform. Experimental results show that the proposed scheme can effectively save the energy consumption of the equipment.**

***Index Terms*—Energy-efficient, mobile cloud computing, multimedia Processing, principal component analysis, nearest neighbor classifier**

I. Introduction

Cloud computing has been proposed as an efficient and cost effective way of providing highly scalable and reliable infrastructures and services to facilitate people's daily activities. In recent years, mobile intelligent terminal and networks, especially 4G networks are developed, which accelerates the emergence of Mobile Cloud Computing (MCC) [1]. Based on MCC, various novel services and applications are available for mobile users, including computing and storage services, mobile social networks, mobile cloud games [2]–[4], and mobile cloud medical treatments. Through MCC-based services and applications, users can access rich resources in cloud anytime and anywhere. Furthermore, as a complementary of limited resources on mobile terminals, MCC can also help significantly improve the Quality of Service (QoS) and Quality of Experience (QoE) [5], [6]. But the existing and related system architecture cannot make full use of the excellent characteristics of MCC. Hence, traditional and resources-intensive services can be re-designed by taking advantages of MCC for mobile users, especially multimedia services.

In this paper, we propose a Cloud-assisted Green Multimedia Processing (CGMP) architecture to achieve energy efficiency of multimedia service for mobile users. MCC-based CGMP offloads computation-intensive tasks to cloud, in order to save energy of terminals. Moreover, based on CGMP, we design a face recognition application with improved Principal Component Analysis (PCA) and nearest neighbor classifier. The experimental results show that CGMP can effectively save the energy consumption. The following summaries key contributions:

- We propose CGMP to offload resource-intensive tasks to cloud for saving the energy consumption of the mobile terminals.
- We design a Hadoop-based multimedia processing framework to relieve bottlenecks of limited resource. By implementing traditional multimedia processing library in parallel and distributed computing, the CGMP architecture can greatly improve the efficiency of multimedia processing.
- We implement a face recognition algorithm based on improved PCA on Hadoop to extract image features and eigenvectors as the input of Linear Discriminant Analysis (LDA) algorithm, and classify using the nearest neighbor classifier.
- The experimental results indicates that MCC-based CGMP can achieve face recognition task in saving mobile device's energy manner. The CGMP architecture can integrate with more mobile applications and be extended to more other application scenarios.

The rest of the paper is organized as follows. In Section 2 we review the related research. Section 3 introduces multimedia processing architectural based on Hadoop cloud platform and OpenCV computer vision library. In Section 4 we describe the framework of face recognition, the procedure and algorithm of recognition, e.g., experimental results and analysis. We end our paper with our conclusions in Section 5.

II. Related Work

In recent years, various categories of services and applications are developed for mobile users, such as entertainment, health, games, social networking, travel and news. However, limited energy of mobile terminals is the main bottleneck for improvement of QoS and QoE [7].

Extending the battery lifetime of mobile devices has became more important, thus several solutions have been proposed to

Min Chen is the corresponding author of this paper.

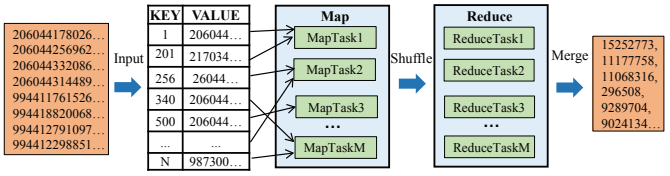

Fig. 1. Hadoop data process flow

enhance CPU performance and to manage other components in an intelligent manner to reduce power consumption [8]. However, these solutions require changes of the structure in mobile devices, or innovations of hardware that result in an increasing cost and may not be feasible for all mobile devices. Computation offloading [9] technique is proposed to migrate the large computations and complex processing from resource-constraint mobile devices to resourceful machines (i.e., servers in clouds) that conserve the battery life [10].

Meanwhile, image processing has been developing very fast and widely, because of its high practical value, which has been used in the fields such as military technology, governmental departments, health care. [11]. However, the traditional image processing technology is not fully suitable for implementing on mobile devices, because of high resource usage of image processing, which can greatly reduce the battery lifetime of mobile terminals [12].

A. Hadoop-based multimedia processing

Hadoop is the most important big data processing platform, which consists of Hadoop Distributed File System (HDFS) and parallel distributed computing framework (MapReduce) [13]. The input/output data of every MapReduce process is a key-value pair. The MapReduce framework divides the input data into multiple data sub-blocks and then activates many MapTasks and/or ReduceTasks for executing parallel data processing. The detailed process is illustrated in Fig. 1. Although, hadoop is mainly utilized to analyze the massive text dataset such as web crawler log and E-commerce log, its recent applications has been used to process other format data [14]. In [15] Wiley *et al.* propose a astronomical image processing system with Hadoop. In their proposal, sequence files group many small files into a few large files to be processed by MapReduce distributed computing model. In [16], Zhang *et al.* established a distributed image retrieval system, in which images are retrieved in a content-based way, and the retrieval among massive image data storage is speeded up by utilizing MapReduce computing framework.

HDFS is designed to read/write large size files, but not optimized for small size files. This is one of the main challenges utilizing hadoop for multimedia processing, because of the image file is mostly small size. Generally, there are the following three approaches to merge small-size files to a large-size file on Hadoop:

- SequenceFile, is a Hadoop file type to store binary key-value pairs, and combine a large number of small-size files [13]. SequenceFile is one of the most common ways to deal with small files on HDFS. Many small files are packed into a single large-size file containing small-size files as indexed elements in key-value pair. Key is the index information of small files, and value is the file data. Although this approach can improve performance of HDFS, input images do not preserve their image formats after merging, and preprocessing is also required for each input images.
- CombineFileInputFormat, is used to merge many files as an input of maptask. Its main idea is to decompose large files or merge small files as multiple input of map. It needs to set three parameters, respectively the smallest split size on a node, the smallest split size on a rack, and the maximum size of a split [17].
- Processing module of small files, is added into HDFS to resolve the small file problem. When a file arrives, the module judges whether the file belongs to a small file. The small files are delivered to the small file processing module, otherwise, delivered to general document processing module [18].

B. Face recoginition

Face recognition can be divided into two types: statistic and geometric methods. Eigenface method is a presentive recognition method based on principal component analysis (PCA) [19], which is widely used in practice. Two-dimensional Principal Component Analysis (2DPCA) is another effective method in the field of pattern recognition. 2DPCA is the improved PCA algorithm, and its major contributions include the construction method of covariance matrix and the selection of maximum eigenvalues [20]. Compared with the traditional PCA algorithm, 2DPCA algorithm is more efficient in the treatment methods of 2D matrix [21]. It does not require that the image must be converted to a vector, but the original image matrix can be directly used to the reconstructed image of the covariance matrix, making the eigenvector's computation of highly efficiency [22].

In contrast to the existing literature where the Hadoop platform is used for a large number of small size image file storage and search, we propose a green and energy saving image processing architecture based on MCC [23]. The computation-intensive tasks are offloaded to cloud to save energy of mobile terminals. Furthermore, we propose a face recognition system using 2DPCA based on MapReduce parallel distributed of hadoop.

III. Green Multimedia Processing Architecture based on Hadoop

We propose a green and energy saving efficient multimedia processing architecture based on Hadoop shown in Fig. 2, which integrates MCC with traditional digital multimedia processing technology. The smart mobile terminal is used for image acquisition and simple preprocessing, whereas the cloud is handling more complex and high energy consuming tasks which are offloaded from mobile terminals. Face recognition is a typical application of CGMP: after capturing a picture, a smart phone will first makes some pretreatments such as image compression, noise reduction. and then will send the picture

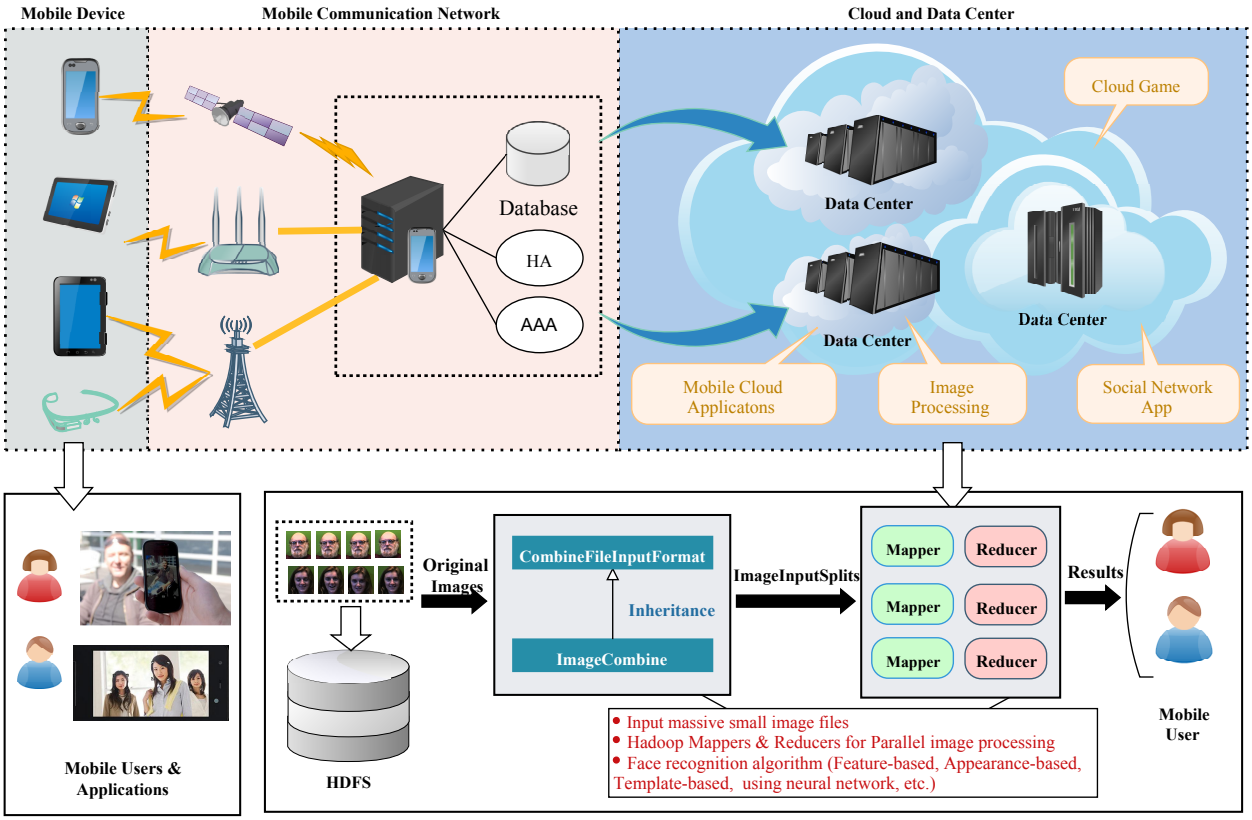

Fig. 2. Green multimedia processing architecture based on Hadoop

to the cloud to match the face against massive face images in the database and running face recognition algorithm. Finally, the cloud returns the recognition result back to the end-user.

We propose a massive image data processing approach based on Hadoop and OpenCV [24] as shown in Fig. 2. We develop *ImageCombine* class derived from *CombineFileInputFormat* to create *ImageInputSplits* as set of images. *ImageCombine* can combine multiple image files and generate *ImageInputSplits* as an input of maptask. In addition, *ImageCombine* selects files which are in the same node to be combined as *ImageInputSplits*. The procedure of image processing is shown below:

1) Move all the images which are needed to be processed into a folder "ImageInput". The image formats can include JPG, BMP, GIF, PNG;
2) Use *ImageCombine* class for image processing, while multiple small-size image files merged into one *ImageInputSplits*;
3) Set all generated *ImageInputSplits* as the input of Hadoop;
4) Task is assigned by the parallel distributed computing framework MapReduce;
5) The input image is processed by JavaCV, which would invoke OpenCV's function library;
6) Output the processing results.

We use the following method to test the multimedia processing platform. At first, we upload multiple small-size images to HDFS, then images will be merged into ImageInputSplits and uploaded ImageInputSplits to the input. Second, we gray the image data of input through OpenCV. Finally, histogram equalization is performed on the gray processing result. Above experiment results prove that the proposed multimedia processing framework based on Hadoop and OpenCV is feasible.

IV. FACE RECOGNITION BASED CGMP

We also design a face recognition system based on CGMP, wich consists of the following five modules: 1) face detection, 2) preprocessing, 3) feature extraction, 4) training and 5) face recognition. The main difference between our system and the traditional face recognition is follow: 1) Using mobile devices acquire human face image in real time, it can be applied to wider areas, 2) Our system can achieve higher performance with the face recognition algorithm of implementation in parallel manner. These modules work as follow steps:

1) *Human face detection* is essential to distinguish the face region from complicate background to minimize the interference information of identification process, in order to improve the efficiency of recognition system. In the proposed system, we use mobile device to extract face image in real time.
2) *Feature extraction* is the process running on cloud to extract eigenvectors from detected face sent by mobile devices.
3) *Multimedia processing and face recognition* is the final procedure to recognize face according to the comparison between the sample and local database.

The algorithm of the face recognition system is based on 2DPCA combined with LDA including two stages, i.e.,

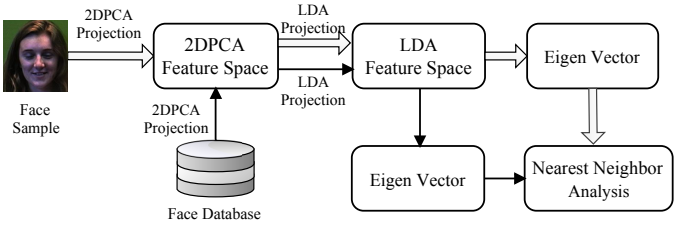

Fig. 3. Face Recognition by 2DPCA combined with LDA

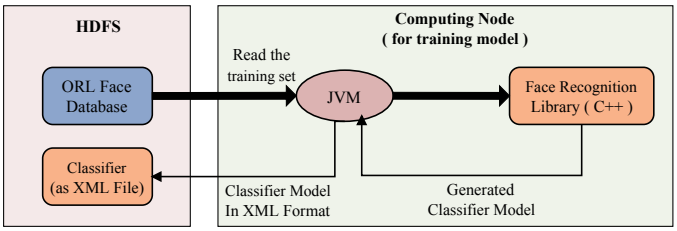

Fig. 4. Training Face Recognition Model

training and identification. Fig. 3, it shows the flow chart of the proposed algorithm.

A. The procedure of training

Face recognition based on Hadoop needs to solve the following two challenges: 1) Classifier is stored in the local system or HDFS and how to read in the classifier, 2) In the recognition stage, how to convert face recognition tasks into $<key, value>$ form for Maptask processing. For the convenience of the compute nodes access, face libraries are stored in HDFS. In the training phase, the nodes read face library from HDFS, and then execute the training mission by the face recognition dynamic library using Java Native Interface (JNI). Finally, the generated classifier is saved to HDFS in XML format. Details are shown in Fig. 4.

The procedure of training model includes the following five steps. Implementation details are shown in Algorithm 1.

Step 1: Set the original training sample is $\left\{S_j^i \in R^{m \cdot n},\ i = 1, 2, \cdots, N;\ j = 1, 2, \cdots, K\right\}$, wherein S_j^i represents the j^{th} image of the i^{th} individual, and M represents the sum of samples and $M = NK$. $\overline{S}$ is the sample mean of all image samples and calculated as Equation 1.

$$\overline{S} = \frac{1}{M} \sum_{i=1}^{N} \sum_{j=1}^{K} S_j^i \tag{1}$$

The connivance matrix G is derived from Equation 2.

$$G = \frac{1}{M} \sum_{i=1}^{N} \sum_{j=1}^{K} \left(S_j^i - \overline{S}\right)^T \left(S_j^i - \overline{S}\right) \tag{2}$$

Step 2: With calculated Eigen value of covariance matrix, we can choose the top p largest eigen values corresponding to orthogonal eigenvectors as projection space $X = [x_1, \cdots, x_p]$. The samples S_j^i are projected to the space as Equation 3 (Y_j^i is the projected matrix).

$$Y_j^i = [S_j^i x_1, \cdots, {S_j}^i x_p] \in R^{m*p} \tag{3}$$

Algorithm 1 2DPCA + LDA Training Procedure

```
//2DPCA training
Read FaceDatabase to FaceModelData[N]
while true
    Generate_Covariance_Matrix(
        FaceModelData[CurrentFaceModel_Index], Covariance_Matrix);
    CurrentFaceModel_Index++;
    if CurrentFaceModel_Index = N then break;
    else continue;
Call_JNI_2DPCA_Calculate(EigenVectorsSpace, Covariance_Matrix);
Call_SaveXML_EigenVector_To(EigenVectorsSpace, HDFS_URL);
//LDA training
while true
    Call_Projection(FaceModelData[Projection_Index],
        EigenVectorsSpace, Projection_Result);
    Projection_Index++;
    if Projection_Index = N then break;
    else continue ;
while true
    Generate_LDA_Covariance_Matrix(
        Projection_Result[LDA_Index],
        LDA_Covariance_Matrix);
    LDA_Index ++;
    if LDA_Index = N then break;
    else continue;
Call_JNI_LDA_Calculate(EigenVectorsSpace2, LDA_Covariance_Matrix, Projection_Result2);
Call_SaveXML_EigenVector_To(EigenVectorsSpace2, HDFS_URL);
```

Step 3: Calculate the mean L_i of all the projected matrixes of individual i as Equation 4:

$$L_i = \frac{1}{K} \sum_{j=1}^{k} Y_j^i \tag{4}$$

Step 4: Take $\left\{Y_j^i \in R^{m*p}, i = 1, 2, \cdots, N;\ j = 1, 2, \cdots, K\right\}$ as the training samples of the traditional LDA algorithm. The between-class covariance matrix G_1 shown in Equation 5 and in-class covariance matrix G_2 are shown in Equation 6.

$$G_1 = \sum_{i=1}^{N} \sum_{j=1}^{K} \left(Y_j^i - L_i\right)\left(Y_j^i - L_i\right)^T \tag{5}$$

$$G_2 = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (L_i - L_j)(L_i - L_j)^T \tag{6}$$

Step 5: We can find out the best projection vectors according to Fisher principle via the feature decomposition method as Equation 7. w_i represents the i eigenvector and λ is the corresponding eigenvalue.

$$G_2 w_i = \lambda G_1 w_i (i = 1, 2, \cdots, N) \tag{7}$$

We can select the eigenvectors corresponding to the top q eigenvalues as the projection space $W = [w_1, w_2, ..., w_q]$, and

the training data $Z^i = [Y_1^i, Y_2^i, \cdots, Y_K^i]$ is projected to the space as Equation 8.

$$\Omega^i = (Z^i)^T W \tag{8}$$

the Ω^i is the eigenvector projected from Z^i to space W.

B. The procedure of recognition

The input file of CGMP is a text file where each line represents a face image address (e.g. *hdfs://namenode-servername/orl/face/1*). The detailed process is illustrated in Fig. 5. By default, the key in the key/value pair is each row of data input data relative to the file offset of the starting character position. In order to facilitate subsequent operations, preprocessing operation converts the default key into the serial number of the image. In the Map phase, MapTasks first according to the value read from the corresponding face image, then call native face recognition using JNI libraries (c++ implementation), start the process of face recognition. Face recognition process reads training model generated at training phase from HDFS to complete the recognition task. In the original scheme, each face image recognition process needs to call the local library using JNI. This needs extra time overhead (as show in Fig. 5 Scheme1). Therefore, we introduce the following improvement scheme, for the first time in execution JNI operation, local face recognition library is cached into memory, then subsequent face recognition tasks on the same node can utilize the cached face recognition library.

The procedure of recognition includes the following three steps. Implementation details are shown in Algorithm 2.

Step 1: The test sample B is projected to the feature space X as Equation 9. C is the projected matrix of B.

$$C = B^T X \tag{9}$$

Step 2: The best projection matrix can be calculated according to Fisher principle, and C is going to project into feature space W as Equation 10.

$$D = C^T W \tag{10}$$

Step 3: Apply the nearest neighbor analysis to find the most proximal projection matrix Ω^i in the matrix D. Ω^i is the outcome of face recognition.

V. ANALYSIS AND EXPERIMENTS

We use CentOS Linux, Apache Hadoop, Oracle Java Development Kit, OpenCV and Eclipse to build the testbed software system and development environment (as summarized in Table I). Our data center hardware includes 7 DataNodes and 2 NameNodes, and each DataNode has the same hardware and software architectures. Computing and storage capacity of data center is shown below : DataNodes (84 cores CPU = 6 cores/CPU * 2 CPUs/Server * 7 Servers, RAM 336GB = 48GB/Server * 7 Servers, Storage 252T = 36T/Server * 7 Servers), NameNode (64 cores CPU = 8 cores/CPU * 4 CPUs/Server * 2 Servers, RAM 256GB = 128GB/Server * 2 Servers, Storage 3.6T = 1.8T/Server * 2 Servers).

Algorithm 2 Face Recognition Procedure

1: **Read face image list to** $FaceList[M]$
2: **while** *true*;
3: Make_Pair(FaceList[index], Pair<key,value>)
4: **if** $index > M$ **then** break;
5: **else** continue;
6: ReadXML_EigenVector(EigenVectorsSpace, EigenVectorsSpace2, HDFS_URL);
7: Call_JNI_Project_To_2DPCA(Pair, EigenVectorsSpace, Result1);
8: Call_JNI_Project_To_LDA (Result1, EigenVectorsSpace2, Result2);
9: **Compare Result2 with Projection_Result2:**
10: **if** $Distance\_between(Result2, Projection\_Result2) = Minimum$ **then**
11: Final_Result=Project_Result2.persion;
12: **else** goto compare;

TABLE I
SOFTWARE SYSTEM AND DEVELOPMENT ENVIRONMENT

Software Tool	Version
Operating System	CentOS 6.3 x86-64 bit
Apache Hadoop	Hadoop-0.23.11
Oracle Java Development Kit	jdk-6u43 x64 for Linux
OpenCV	OpenCV-2.4.9
Eclipse IDE for Java Developers	Eclipse Luna (4.4.1) for Linux 64 bit

The experimental data is ORL [25] face database including a number of face pictures with various emotions. In the ORL standard face database, we choose ten images of each type for training samples, while another ten images for testing. This section mainly focuses on the comparison of accuracy of face recognition among these algorithms: 2DPCA combined with LDA, PCA, 2DPCA and LDA (as shown in Fig. 6.).

The Euclidean distance is the distance between the samples

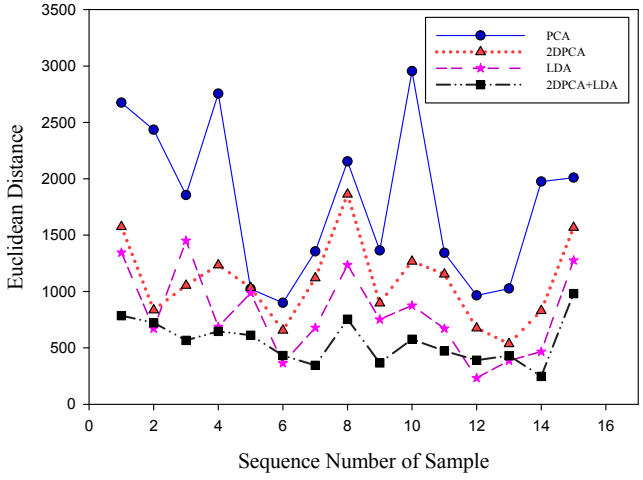

Fig. 6. Euclidean distance comparison of algorithms

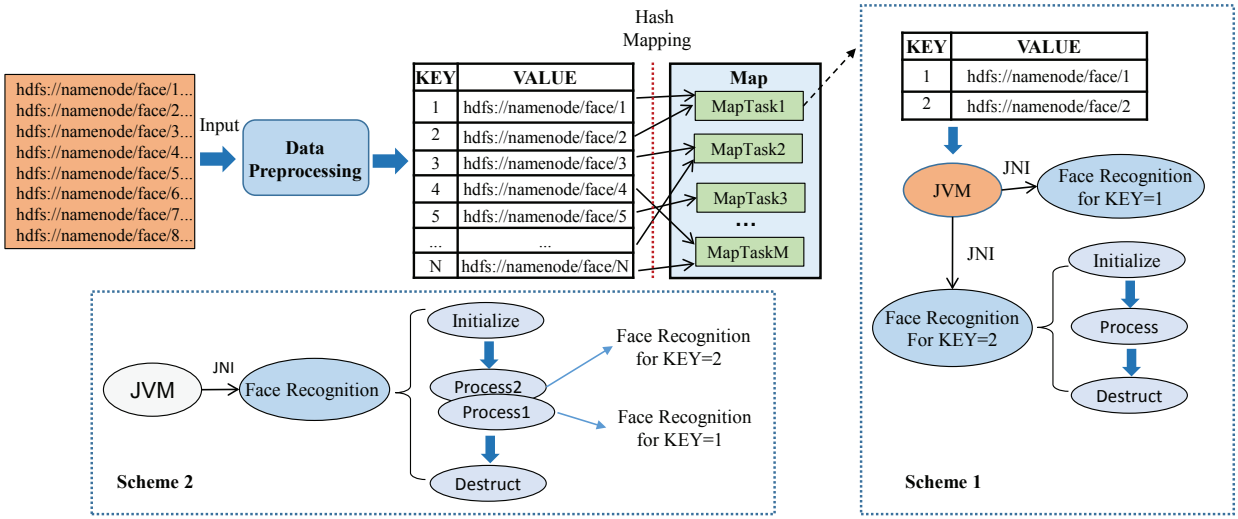

Fig. 5. Parallel face recognition based on CGMP

and the templates, which means the error range of the recognition result. Thus, we assume that the smaller the Euclidean distance is the better accuracy of the recognition is achieved. 2DPCA algorithm is used for principal component analysis based on two dimensional image data, which is extended for improving the calculation and feature extraction. By analyzing the Euclidean distance we prove that the smaller the Euclidean distance is, the better of the recognition result is achieved. As we can be seen from the Fig. 6, 2DPCA has smaller Euclidean distance than PCA in our experimental results.

The LDA algorithm makes the samples to be projected to the lower dimensional space which makes the dispersion of in-class of all samples to be minimum and the dispersion of between-class to be maximum. Through the experiments, it is obvious that the face recognition based on LDA and PCA are accurate, but the Euclidean distance of LDA is smaller than PCA. But the LDA is not better than PCA in some other situation. Traditional LDA method's accuracy rate of face recognition relies on the rank of between-class scatter matrix, because the rank restrains the number of eigenvectors. When the rank R of the between-class scatter matrix comes to small, that means only R's distinguishing direction could work on the process of face recognition. Meanwhile, LDA algorithm overemphasizes the distance between the edge classes and other classes, if the amount of samples is greater than the dimension of these samples while a sample is projected onto the optimal projection direction. Samples of nearest neighbor classes occurs overlapped would influence the procedure of calculating the eigenvectors of in-class scatter matrix, and that causes the small-sample problem.

The numerical comparison of the four algorithms is shown in Table II. The experimental results verify that only in two samples the performance of 2DPCA+LDA falls behind the LDA, while in the other tests it exceeds the LDA.

Analyzing the reason, we conclude that according to the principle of Fisher can easily lead to the small-sample problem, and it's too hard to find enough trained samples to ensure the invertibility of in-class scatter matrix, that result in loss of many useful identifiable information, which restrains the farther recognition.

TABLE II
COMPARISION OF SERVAL FACE RECOGNITION ALGORITHMS

Algorithm	Average Euclidean Distance	Recognition Rate
PCA	2354	86%
LDA	865	95%
2DPCA	1123	90%
2DPCA + LDA	486	96.5%

2DPCA+LDA sustains high rate of recognition, and solves the small-sample problem. As for the dimensionality reduction of higher space, it avoids the invertibility of in-class scatter matrix, then persist the identifiable useful information. Because 2DPCA+LDA take advantages of both of 2DPCA and LDA, 2DPCA is used for principal component extraction, LDA lessen the edge class's leadership to eigen decomposition , making the identifiable information more clear. Thus the algorithm of 2DPCA combined LDA can get a better rate of recognition than others.

VI. CONCLUSION

We focused on energy consumption of multimedia processing technology in mobile cloud computing, and proposed an approach to assign mobile devices with simple tasks while offload complex tasks to the cloud. Furthermore, we designed a CGMP-based face recognition system using algorithm of 2DPCA combined with LDA. The experimental results show the advantages of the CGMP platform in processing face recognition applications. Our long-term goal is to propose an architecture of CGMP to support all new applications with multimedia function.

REFERENCES

[1] L. Lei, Z. Zhong, K. Zheng, J. Chen, and H. Meng, "Challenges on wireless heterogeneous networks for mobile cloud computing," *Wireless Communications, IEEE*, vol. 20, no. 3, pp. 34–44, 2013.

[2] D. Wu and Z. Xue, "Cloud Gaming: From Concept to Reality," *IEEE COMSOC MMTC E-Letter, Special Issue on Cloud Computing for Multimedia"*, vol. 8, no. 6, November 2013.

[3] D. Wu, Z. Xue, and J. He, "iCloudAccess: Cost-effective streaming of Video Games from the Cloud with Low Latency," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 8, pp. 1405 – 1416, Aug. 2014.

[4] H. Hu, Y. Wen, H. Luan, T. Chua, and X. Li, "Towards multi-screen social tv with geo-aware social sense," *IEEE MultiMedia, Feb*, 2014.

[5] K. Zheng, X. Zhang, Q. Zheng, W. Xiang, and L. Hanzo, "Quality-of-experience assessment and its application to video services in lte networks," *Wireless Communications, IEEE*, vol. 22, no. 1, pp. 70–78, 2015.

[6] Y. Wen, X. Zhu, J. Rodrigues, and C. W. Chen, "Cloud mobile media: Reflections and outlook," *Multimedia, IEEE Transactions on*, vol. 16, no. 4, pp. 885–902, June 2014.

[7] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya, "Cloud-based augmentation for mobile devices: Motivation, taxonomies, and open challenges," *Communications Surveys Tutorials, IEEE*, vol. 16, no. 1, pp. 337–368, First 2014.

[8] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*. USENIX Association, 2010, pp. 4–4.

[9] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *Wireless Communications, IEEE Transactions on*, vol. 12, no. 9, pp. 4569–4581, 2013.

[10] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, 2010.

[11] M. Sonka, H. Vaclav, and R. Boyle, "Image processing, analysis, and machine vision. cengage learning, 2008," ISBN 978-0-495-24438-7. International Student Edition, Tech. Rep.

[12] T. Ma, M. Hempel, D. Peng, and H. Sharif, "A survey of energy-efficient compression and communication techniques for multimedia in resource constrained systems," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 3, pp. 963–972, 2013.

[13] T. White, *Hadoop: The definitive guide*. " O'Reilly Media, Inc.", 2012.

[14] M. Chen, S. Mao, and Y. Liu, "Big data: A survey," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, 2014.

[15] K. Wiley, A. Connolly, J. Gardner, S. Krughoff, M. Balazinska, B. Howe, Y. Kwon, and Y. Bu, "Astronomy in the cloud: using mapreduce for image co-addition," *Astronomy*, vol. 123, no. 901, pp. 366–380, 2011.

[16] J. Zhang, X. Liu, J. Luo, and B. Lang, "Dirs: Distributed image retrieval system based on mapreduce," in *Pervasive Computing and Applications (ICPCA), 2010 5th International Conference on*. IEEE, 2010, pp. 93–98.

[17] D. Borthakur, "HDFS architecture guide," *HADOOP APACHE PROJECT http://hadoop. apache. org/common/docs/current/hdfs design. pdf*, 2008.

[18] B. Dong, J. Qiu, Q. Zheng, X. Zhong, J. Li, and Y. Li, "A novel approach to improving the efficiency of storing and accessing small files on hadoop: a case study by powerpoint files," in *Services Computing (SCC), 2010 IEEE International Conference on*. IEEE, 2010, pp. 65–72.

[19] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[20] J. Pei, Y. Huang, X. Liu, J. Yang, Z. Cao, and B. Wang, "2DPCA-based two-dimensional maximum interclass distance embedding for sar atr," in *Communications, Circuits and Systems (ICCCAS), 2013 International Conference on*, vol. 1. IEEE, 2013, pp. 267–270.

[21] J. H. Shah, M. Sharif, M. Raza, and A. Azeem, "A survey: Linear and nonlinear pca based face recognition techniques," *International Arab Journal of Information Technology*, no. 10, pp. 536–545, 2013.

[22] S.-M. Hu, T. Chen, K. Xu, M.-M. Cheng, and R. R. Martin, "Internet visual media processing: a survey with graphics and vision applications," *The Visual Computer*, vol. 29, no. 5, pp. 393–405, 2013.

[23] H. Hu, Y. Wen, T.-S. Chua, and X. Li, "Toward scalable systems for big data analytics: A technology tutorial," *Access, IEEE*, vol. 2, pp. 652–687, 2014.

[24] W. Garage, "Opencv," *Open Source Computer Vision Library*, 2011.

[25] C. University, "ORL database of faces," http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html, 2002.